A MULTIGRID ALGORITHM FOR THE CELL-CENTERED FINITE DIFFERENCE SCHEME*

Richard E. Ewing and Jian Shen
Institute for Scientific Computation
Texas A&M University
College Station, Texas

SUMMARY

In this article, we discuss a non-variational V-cycle multigrid algorithm based on the cell-centered finite difference scheme for solving a second-order elliptic problem with discontinuous coefficients. Due to the poor approximation property of piecewise constant spaces and the non-variational nature of our scheme, one step of symmetric linear smoothing in our V-cycle multigrid scheme may fail to be a contraction. Again, because of the simple structure of the piecewise constant spaces, prolongation and restriction are trivial; we save significant computation time with very promising computational results.

INTRODUCTION

In the simulation of incompressible fluid flow in porous media, we have to solve at least one second-order elliptic equation per each time step. A very important quantity is the Darcy velocity, defined by

$$\mathbf{u} = -\mathcal{K}\nabla p \tag{1}$$

where p is the pressure of the fluid and $\mathcal{K}$ is the conductivity. $\mathcal{K}$ can be written by $\mathcal{K} = \frac{k}{\mu}$, where k is a tensor representing the permeability of the medium which can be discontinuous in general, and μ represents the viscosity of the fluid. μ is a continuous function of both time and space variables, but may have a very sharp frontal change of values. In other words, μ can change rapidly inside the interesting domain and the region of rapid change may move as time changes. According to the conservation law of mass balance, the Darcy velocity $\mathbf{u}$ must be continuous along the normal direction at an element or domain boundary, no matter whether $\mathcal{K}$ is discontinuous or not.

Now, we consider the following simple second-order elliptic equation in mixed form. Find a pair $(p, \mathbf{u})$ such that

$$\begin{aligned} \mathbf{u} &= -\mathcal{K}\nabla p, && \text{in } \Omega = (0,1)^2 \subset \mathbb{R}^2, \\ \nabla \cdot \mathbf{u} &= f, && \text{in } \Omega, \\ p &= 0, && \text{on } \partial\Omega, \end{aligned} \tag{2}$$

*This work was supported in part by the Department of Energy under Contract No. DE-FG05-92ER25143. The authors would also like to thank Joe Pasciak for valuable discussions about this work.

where the conductivity $\mathcal{K}(x,y) = \mathrm{diag}(a,b)$ is positive and uniformly bounded above and below.

Because of the discontinuity of $\mathcal{K}$, the classical solution of p in (2) may not exist. Let $(\cdot,\cdot)$ denote $L^2(\Omega)$ or $(L^2(\Omega))^2$ inner product and $H(\mathrm{div};\Omega) \equiv \left\{\mathbf{u} \in (L^2(\Omega))^2 \mid \nabla\cdot\mathbf{u} \in L^2(\Omega)\right\}$. We seek the solution pair $(p,\mathbf{u}) \in H^1(\Omega) \times H(\mathrm{div};\Omega)$, such that

$$\begin{aligned} (\mathcal{K}^{-1}\mathbf{u},\mathbf{v}) &= (p,\nabla\mathbf{v}), \quad \forall\, \mathbf{v} \in H(\mathrm{div},\Omega), \\ (\nabla\cdot\mathbf{u},w) &= (f,w), \qquad w \in L^2(\Omega). \end{aligned} \tag{3}$$

In [5], error estimates for solving (3) by the cell-centered finite difference scheme are studied, with the following results:

$$\|P - \mathcal{P}p\|_{L^2} + \|\mathbf{U} - \pi\mathbf{u}\|_{L^2} \le ch^s \|p\|_{1+s,\Omega\setminus\Gamma}, \qquad s = 1,2, \tag{4}$$

where $\mathcal{P}\times\pi$ is the Raviart-Thomas projection, Γ are the lines of discontinuity which coincide with the grid lines, and $(P,\mathbf{U})$ is the numerical solution of the cell-centered finite difference to approximate (3)[5]. Actually, we view the cell-centered finite difference method as a special numerical integration of the Raviart-Thomas mixed finite element method [4–6]. For $s = 2$, (4) is the superconvergence error estimate.

From the point of view of mass balance and accuracy, the cell-centered finite difference scheme is one of the best numerical schemes to fulfill our goal. In this article, we investigate the efficiency of the multigrid algorithm based on the cell-centered finite difference scheme introduced in [5].

NUMERICAL SCHEME IN MULTIGRID SETTING

Let us use the Laplacian operator, $-\Delta$, to explain the cell-centered finite difference scheme stencil. For an interior node, the stencil for $-\Delta$ is (a) in Figure 1. For a corner node, the stencil for $-\Delta$ is (b) in Figure 1. For other boundary nodes, the stencil for $-\Delta$ is (c) in Figure 1. For discontinuous conductivity, see [5] for details. Now, we consider the uniform grid only. Let $\mathcal{M}_k$ denote the piecewise constant Raviart-Thomas rectangular pressure space defined on Ω with mesh size $h_k = 2^{-(k+1)}$, $k = 0,1,2,3,\ldots,J$. It is clear that

$$\mathcal{M}_0 \subset \mathcal{M}_1 \subset \mathcal{M}_2 \subset \ldots \subset \mathcal{M}_{J-1} \subset \mathcal{M}_J \subset L^2(\Omega). \tag{5}$$

With an abuse of notation, for $u \in \mathcal{M}_k$, u is either a piecewise function or a vector with its nodal values as its entries. On $\mathcal{M}_k$, the cell-centered finite difference approximation is to find $P \in \mathcal{M}_k$, such that

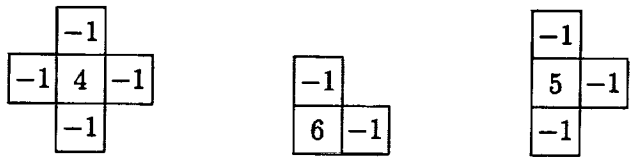
$$\tilde{A}_k P = F_k \equiv \mathcal{P}_k f, \qquad k = 0,1,2,\ldots,J. \tag{6}$$

Here $\mathcal{P}_k : L^2 \to \mathcal{M}_k$ is the L^2-projection into $\mathcal{M}_k$ defined by

$$(f,w) = (\mathcal{P}_k f, w), \qquad \forall\, w \in \mathcal{M}_k, \tag{7}$$

and f is the load function of (2). The corresponding stencil of $\tilde{A}_k$ is shown in Figure 1. Our goal is to find $P \in \mathcal{M}_J$, such that

$$\tilde{A}_J P = F_J = \mathcal{P}_J f. \tag{8}$$



(a) (b) (c)

FIGURE 1. *Stencils for the Laplacian operator.*

The discrete L^2-inner product and associated norm on $\mathcal{M}_k$ are denoted by

$$(u,v)_k = h_k^2 v^T u \quad \text{and} \quad \|u\|_k^2 = (u,u)_k, \qquad u,v \in \mathcal{M}_k, \tag{9}$$

where $v^T u$ is the usual algebraic inner product. Let $A_J = \tilde{A}_J$ define an associated bilinear form A_J on $\mathcal{M}_J$ by

$$(A_J w, \phi)_J = A_J(w,\phi), \qquad \forall\, w,\phi \in \mathcal{M}_J. \tag{10}$$

Before we define A_k for $0 \le k < J$, we first define the prolongation operator I_k and the restriction operator P_{k-1}^0. Let $I_k : \mathcal{M}_{k-1} \to \mathcal{M}_k$, $k = 1,2,\ldots,J$ be the natural imbedding from $\mathcal{M}_{k-1}$ to $\mathcal{M}_k$. Thus $P_{k-1}^0 : \mathcal{M}_k \to \mathcal{M}_{k-1}$, the adjoint of I_k in $(\cdot,\cdot)_k$, is defined by

$$(P_{k-1}^0 w, \phi)_{k-1} = (w, I_k \phi)_k, \qquad w \in \mathcal{M}_k,\ \phi \in \mathcal{M}_{k-1}. \tag{11}$$

From (9) and $h_{k-1} = 2h_k$, it is clear that $P_{k-1}^0 = \frac{1}{4} I_k^T$ in matrix form. Now, we define the bilinear form $A_{k-1}(\cdot,\cdot)$ and the matrix A_{k-1} on $\mathcal{M}_{k-1}$ for $k = J, J-1, \ldots, 2, 1$, by

$$2A_{k-1}(u,v) = A_k(I_k u, I_k v), \qquad \forall\, u,v \in \mathcal{M}_{k-1}, \tag{12}$$

and the corresponding matrix relation is

$$A_{k-1} = \frac{1}{8} I_k^T A_k I_k = \frac{1}{2} P_{k-1}^0 A_k I_k. \tag{12'}$$

Remark 1. It is shown in [5] that for piecewise smooth conductivity tensor $\mathcal{K}$, as long as the discontinuities coincide with the coarser grid lines

$$A_{k-1} = \left(1 + O(h_k^2)\right) \tilde{A}_{k-1}. \tag{13}$$

In (13) $O(h_k^2) = Ch_k^2$. C depends on the local smoothness of $\mathcal{K}$ but is independent of the jumps. Since I_k is a simple operator, it is much easier to generate A_{k-1} by (12') than by (6) directly. Of course, A_k, $k = 0,1,2,\ldots,J-1$, are all positive definite since A_J is, and the spaces are nested. Because of (12), our multigrid algorithm can be considered as a black box solver once I_k has been defined. We mention that (12) holds for three-dimensional problems of $-\nabla \cdot (\mathcal{K}\nabla u)$, with (12') being changed to

$$A_{k-1} = \frac{1}{16} I_k^T A_k I_k = \frac{1}{2} P_{k-1}^0 A_k I_k.$$

We also define the adjoint of I_k in $A_k(\cdot,\cdot)$, $P_{k-1} : \mathcal{M}_k \to \mathcal{M}_{k-1}$ by

$$A_{k-1}(P_{k-1}u, v) = A_k(u, I_k v), \qquad u \in \mathcal{M}_k, \quad v \in \mathcal{M}_{k-1}. \tag{14}$$

To define the smoothing process, we require linear operators $R_k : \mathcal{M}_k \to \mathcal{M}_k$ for $k = 1, 2, \ldots, J$. These operators may or may not be symmetric with respect to the inner product $(\cdot,\cdot)_k$. Let $A_k = D_k + L_k + L_k^T$, D_k be the diagonal part of A_k, and L_k be the lower triangular part of A_k. The linear smoothers we have tried are the following relaxation schemes. For $0 < \omega < 2$,

$$\begin{aligned} &\text{(a) Gauss-Seidel:} && R_k = \left(\frac{D_k}{\omega} + L_k\right)^{-1} \text{ and } R_k^T, \\ &\text{(b) Jacobi:} && R_k = \omega D_k^{-1}, \\ &\text{(c) Richardson:} && R_k = \frac{\omega}{\lambda_k} I, \end{aligned} \tag{15}$$

where I is the identity operator on $\mathcal{M}_k$ and λ_k is the spectral radius of A_k. We allow the relaxation parameter ω to be different for pre-smoothing and post-smoothing processes in the following definition.

Following [1] the multigrid operator $B_k : \mathcal{M}_k \to \mathcal{M}_k$ is defined by induction and is given as follows. The pre-smoother is denoted by R_k and the post-smoother by $\bar{R}_k$.

V-Cycle Multigrid Algorithm:

Set $B_0 = A_0^{-1}$. Assume that B_{k-1} has been defined and define $B_k g$ for $g \in \mathcal{M}_k$ as follows:

1. Set $x^0 = 0$.
2. Define x^ℓ for $\ell = 1, 2, \ldots, m(k)$ by $x^\ell = x^{\ell-1} + R_k(g - A_k x^{\ell-1})$.
3. Set $y^0 = x^{m(k)} + I_k B_{k-1} P_{k-1}^0 \left(g - A_k x^{m(k)}\right)$.
4. Define y^ℓ for $\ell = 1, 2, \ldots, m(k)$ by $y^\ell = y^{\ell-1} + \bar{R}_k(g - A_k y^{\ell-1})$.
5. Set $B_k g = y^{m(k)}$.

Remark 2. Since equation (12) holds for all levels, this multigrid algorithm is non-variational according to [1], but the approximation property (4) is valid for each level as long as the non-variational relation (12) is satisfied. In this algorithm $m(k)$ is a positive integer which may vary from level to level. In general this multigrid algorithm is not symmetric in $(\cdot,\cdot)_k$ except for $\bar{R}_k = R_k^T$.

Setting $K_k = I - R_k A_k$ and $\bar{K}_k = I - \bar{R}_k A_k$, it is straightforward to check that

$$\begin{aligned} I - B_k A_k &= \bar{K}_k^{m_2(k)} [I - I_k B_{k-1} P_{k-1}^0 A_k] K_k^{m_1(k)} \\ &= \bar{K}_k^{m_2(k)} [I - I_k B_{k-1} A_{k-1} P_{k-1}] K_k^{m_1(k)}. \end{aligned} \tag{16}$$

Equation (16) gives a fundamental recurrence relation for the multigrid operator B_k.

COMPUTATIONAL EXPERIMENTS

We have tested the multigrid algorithm described in Section 2. We use a power method to compute the largest and the smallest eigenvalues of $B_J A_J$.

The linear smoothers we have tried are the following. Let m be a positive integer. $m(k) = m$ for all k,

$$S_1(m): \quad R_k = \frac{1}{\lambda_k} I, \quad \bar{R}_k = R_k,$$

$$S_2(m): \quad R_k = \frac{1}{\lambda_k} I, \quad \bar{R}_k = 2R_k, \quad \text{where } \lambda_k \text{ is the largest eigenvalue of } A_k,$$

$$S_3(m): \quad R_k = (D_k + L_k)^{-1}, \quad \bar{R}_k = R_k^T,$$

$$S_4(m): \quad R_k = 1.35 D_k^{-1}, \quad \bar{R}_k = \frac{1}{2} R_k,$$

$$S_5(m): \quad R_k = \left(\frac{D_k}{1.35} + L_k\right)^{-1}, \quad \bar{R}_k = \left(\frac{D_k}{0.675} + L_k^T\right)^{-1},$$

$$S_6(m): \quad R_k = \left(\frac{2}{3} D_k + L_k\right)^{-1}, \quad \bar{R}_k = (2D_k + L_k^T)^{-1}.$$

Note that only $S_1(m)$ and $S_3(m)$ make $B_J A_j$ $A_J(\cdot,\cdot)$ symmetric. The rest are neither symmetric nor $A_J(\cdot,\cdot)$ symmetric. We also have tried nonlinear smoothers, conjugate gradient, and diagonally preconditioned conjugate gradient algorithms. We shall use $N(m)$ to represent our nonlinear multigrid by diagonally preconditioned conjugate gradient smoothers. The reason we choose different relaxation numbers comes from the suggestion [3] for an algebraic multigrid algorithm, and from our computational experiments.

We list our test results in Tables 1–9 at the end of this paper for the following problems:

Ex. 1. Poisson problem: $\mathcal{K} \equiv 1$ in (2).

Ex. 2. Isotropic problem with nearly singular piecewise smooth conductivity:

$$\mathcal{K} = \Big[0.001 + 11.1\big(1 + \cos(3.561\pi x)\sin(3.001\pi y)\big)\Big]\, q,$$

$$q = \begin{cases} 10^{-4}, & \text{if } x \geq \frac{1}{2} \text{ and } y \geq \frac{1}{2}, \\ 1, & \text{otherwise.} \end{cases}$$

Ex. 3. Same kind of problems as Ex. 2:

$$\mathcal{K} = \Big[0.001 + 45.1\Big(1 + \cos(9.431\pi x)\sin(3.001\pi y)\Big)\Big]\, q,$$

$$q = \begin{cases} 10^4, & \text{if } x \geq \frac{1}{2} \text{ and } y \geq \frac{1}{2}, \\ 1, & \text{otherwise.} \end{cases}$$

Ex. 4. Anisotropic problem with smooth conductivity:

$$\begin{aligned} \mathcal{K} &= \operatorname{diag}(a, b), \\ a &= 0.001 + 45.1\Big(1 + \cos(9.431\pi x)\sin(9.431\pi y)\Big), \\ b &= 0.001 + 45.1\,(1 + \sin(9.431\pi x)\cos(9.431\pi y))\,. \end{aligned}$$

Note that all the solutions of our examples have the superconvergence results proved in [5], i.e., satisfying (4) with $s = 2$.

In Tables 1 and 6, for example, the second row of Table 1 means $J + 1 = 3$ level multigrid with $h_J = \frac{1}{8}$, $\lambda_m, S_1(1)$ means $\lambda_m = \min \lambda(B_J A_J)$ by $S_1(1)$ smoothers, and $\lambda_M, S_1(1)$ means $\lambda_M = \max \lambda(B_J A_J)$ by $S_1(1)$ smoothers. From Table 1, we can see that even when $I - B_J A_J$ fails to be a reducer, B_J may still be a good preconditioner. In Tables 5–7, it is interesting to see the relations of the number of V-cycles ($\#V$), average contraction numbers (avc) and the time spent on the machine (cpu in seconds) when solving a fixed problem on a fixed grid by using different multilevels. In Tables 3–5, and 7–9, avc is defined by

$$\text{avc} = \frac{1}{n}\sum_{j=1}^{n} \frac{\|r_j\|_J^2}{\|r_{j-1}\|_y^2},$$

where $n = \#V$ is the total number of V-cycles and $\|r_j\|_J$ is the discrete L^2-norm of the residual after the jth V-cycle. The stop tolerance for all the iterative algorithms is $\|r_n\|_J^2 \leq \epsilon = 10^{-14}$. Our coarsest grid solver is a diagonal preconditioned conjugate gradient solver with tolerance $\epsilon_0 = 10^{-19}$. In Tables 7–9, "cg" means the standard conjugate gradient algorithm, its corresponding "$\#V$" means the total iteration steps, when $\|r_n\|_J^2 \leq \epsilon = 10^{-14}$, and "bpcg" means the incomplete factorization preconditioned conjugate gradient algorithm [2].

REFERENCES

1. Bramble, J. H, Pasciak, J. E., and Xu, J.: The Analysis of Multigrid Algorithm with Nonnested Spaces or Noninherited Quadratic Forms. *Math. Comp.*, vol. 56, 1991, pp. 1–34.

2. Concus, P., Golub, G. H., and Meurant, G.: Block Preconditioning for the Conjugate Gradient Method. *SIAM J. Sci. Stat. Comput.*, vol. 6, 1985, pp. 220–252.

3. McCormick, S. F., ed.: *Multigrid Methods.* SIAM, Philadelphia, Pennsylvania, 1987.

4. Russell, T.F. and Wheeler, M.F.: Finite Element and Finite Difference Methods for Continuous Flows in Porous Media. *The Mathematics of Reservoir Simulation* (R. E. Ewing, ed.). SIAM, Philadelphia, Pennsylvania, 1983.

5. Shen, J.: *Mixed Finite Element Methods: Analysis and Computational Aspects.* Ph.D. Thesis, University of Wyoming, 1992.

6. Weiser, A. and Wheeler, M.F.: On the Convergence of Block-Centered Finite Differences for Elliptic Problems. *SIAM J. Numer. Anal.*, vol. 25, 1988, pp. 351–375.

Table 1. For Ex. 1

GRID	J	$\lambda_m, S_1(1)$	$\lambda_M, S_1(1)$	$\lambda_m, S_1(2)$	$\lambda_M, S_1(2)$
4^2	1	0.548	1.351	0.788	1.134
8^2	2	0.446	1.804	0.704	1.297
16^2	3	0.397	2.394	0.663	1.470
32^2	4	0.367	3.128	0.639	1.633
64^2	5	0.345	4.023	0.623	1.783
128^2	6	0.325	5.106	0.609	1.924
256^2	7	0.299	6.417	0.592	2.059

Table 2. For Ex. 1

GRID	J	$\lambda_m, S_3(1)$	$\lambda_M, S_3(1)$	$\lambda_m, S_3(2)$	$\lambda_M, S_3(2)$
4^2	1	0.858	1.142	0.971	1.037
8^2	2	0.812	1.239	0.960	1.062
16^2	3	0.794	1.344	0.954	1.089
32^2	4	0.785	1.445	0.951	1.112
64^2	5	0.784	1.535	0.950	1.131
128^2	6	0.784	1.614	0.949	1.146
256^2	7	0.783	1.685	0.949	1.159

Table 3. For Ex. 1 by $B_J(S_2(1))$

GRID		$J=1$	$J=2$	$J=3$	$J=4$	$J=5$	$J=6$	$J=7$	$J=8$
	$\#V$	23	34	45	48	50	50		
64^2	avc	0.121	0.243	0.349	0.374	0.389	0.389		
	cpu	4.4	2.1	1.7	1.7	1.7	1.7		
	$\#V$	24	38	52	62	69	71	71	
128^2	avc	0.126	0.266	0.384	0.468	0.489	0.500	0.500	
	cpu	35	11.5	7.1	7.0	7.0	7.0	7.0	
	$\#V$	26	40	57	75	92	97	99	99
256^2	avc	0.129	0.27	0.405	0.502	0.572	0.584	0.586	0.586
	cpu	248.0	75.2	35.2	34.3	35.3	35.1	35.1	35.2

Table 4. For Ex. 1 by $B_J(S_3(1))$

GRID		$J=1$	$J=2$	$J=3$	$J=4$	$J=5$	$J=6$	$J=7$	$J=8$
	$\#V$	16	22	28	33	34			
64^2	avc	0.050	0.118	0.185	0.256	0.256			
	cpu	4.0	1.7	1.5	1.5	1.5			
	$\#V$	17	24	31	38	43	46	46	
128^2	avc	0.053	0.129	0.204	0.275	0.325	0.345	0.345	
	cpu	33.0	8.5	7.0	6.0	6.1	6.4	6.4	
	$\#V$	18	26	34	42	51	58	61	61
256^2	avc	0.054	0.136	0.219	0.296	0.367	0.417	0.436	0.436
	cpu	252.0	61.0	27.0	24.0	28.0	31.0	32.0	32.0

Table 5. For Ex. 1 by $B_J(S_3(2))$

GRID		$J=1$	$J=2$	$J=3$	$J=4$	$J=5$	$J=6$	$J=7$	$J=8$
	$\#V$	9	10	11	11	11	11		
64^2	avc	0.0055	0.0092	0.011	0.012	0.0121	0.0121		
	cpu	3.3	2.0	1.5	1.0	1.0	1.0		
	$\#V$	10	11	12	12	12	12	12	
128^2	avc	0.0066	0.011	0.0136	0.015	0.0155	0.0156	0.0156	
	cpu	17.0	5.3	3.8	3.0	3.0	3.2	3.2	
	$\#V$	10	12	12	13	13	13	13	13
256^2	avc	0.0067	0.015	0.015	0.018	0.019	0.0191	0.0191	0.0191
	cpu	152.0	34.0	17.0	16.0	15.0	15.3	15.3	15.3

Table 6. For Ex. 2

GRID	J	$\lambda_m, S_3(1)$	$\lambda_M, S_3(1)$
4^2	1	0.772	1.090
8^2	2	0.687	1.208
16^2	3	0.718	1.329
32^2	4	0.737	1.442
64^2	5	0.751	1.541
128^2	6	0.759	1.626
256^2	7	0.762	1.699

Table 7. For Ex. 3

GRID	J		$N(1)$	$S_3(1)$	$S_5(1)$	$S_4(1)$	cg	$\|r_0\|_J^2$
		$\#V$	25	33	25	34	17,445	
64^2	5	avc	0.164	0.255	0.157	0.276		1.4×10^6
		cpu	2.3	0.6	0.3	0.6	143.0	
		$\#V$	29	45	29	35	55,647	
128^2	6	avc	0.195	0.35	0.202	0.258		1×10^7
		cpu	12.5	4.5	3.5	4.5	1,835.0	
		$\#V$	31	61	33	38	142,610	
256^2	7	avc	0.213	0.449	0.270	0.071		8.2×10^7
		cpu	53.5	27.5	15.5	19.5	17,003.0	

Table 8. For Ex. 3

GRID	J		$N(2)$	$S_3(2)$	$S_5(2)$	$S_6(2)$	$S_4(2)$	bpcg
		$\#V$	12	11	11	15	17	26
64^2	5	avc	0.028	0.016	0.011	0.047	0.071	
		cpu	2.3	1.0	1.0	1.0	1.0	1.2
		$\#V$	14	12	11	17	17	41
128^2	6	avc	0.034	0.020	0.012	0.053	0.061	
		cpu	9.0	1.8	1.6	2.5	3.5	5.5
		$\#V$	14	13	13	18	19	63
256^2	7	avc	0.035	0.023	0.017	0.056	0.071	
		cpu	36.5	11.5	11.5	14.5	17.5	33.5

Table 9. For Ex. 4

GRID	J		$N(3)$	$S_6(2)$	$S_4(3)$	bpcg	cg	$\|r_0\|_J^2$
		$\#V$	13	18	21	27	313	
64^2	5	avc	0.012	0.042	0.084			1.2×10^{10}
		cpu	32.0	0.5	1.5	1.3	2.3	
		$\#V$	16	19	31	40	651	
128^2	6	avc	0.031	0.048	0.181			9.1×10^{10}
		cpu	13.0	2.5	12.0	5.5	23.0	
		$\#V$	21	25	57	62	1,329	
256^2	7	avc	0.07	0.091	0.374			7.2×10^{11}
		cpu	68.0	18.0	64.0	34.0	161.0	